International Journal of Information Systems Management Research and Development (IJISMRD)
ISSN(P): 2250-236X; ISSN(E): 2319-4480
Vol. 4, Issue 2, Apr 2014, 35-48

GEOMETRIC IMAGE EDGE DETECTION TECHNIQUES

NASSAR H. ABDEL-ALL, M. A. SOLIMAN, R. A. HUSSIEN & WADAH M. EL-NINI

Department of Mathematics, Faculty of Science, Assiut University, Assiut, Egypt

ABSTRACT

Edges characterize boundaries are the problem of fundamental importance in description of geometric surfaces. Since edge detection is in the fore front of computer vision system for detection of surfaces images as Revolution and Saddle surfaces needs to description, it is crucial to have a good under standing of geometric properties of Surfaces and edge detection algorithms.

In this paper the comparative analysis of various surface-image edge detection techniques and the study of the geometric properties of Surfaces are presented. The software is developed using MATLAB 7.11. It has been shown that the Canny's edge detection algorithm performs better than all operators (i.e. Robert, Prewitt, Sobel, Zero-cross (Laplacian) and LOG) under almost all scenarios.

KEYWORDS: Edge Detection, Noise, Computer Vision, Revolution Surface, Saddle Surface

INTRODUCTION

Edge detection refers to the process of identifying locating sharp discontinuities in an image like images of different geometric surfaces. The discontinuities are abrupt changes in pixel intensity which characterize boundaries of objects in an image.

Classical methods of edge detection involve convolving the image with an operator (a 2-D filter), which is constructed to be sensitive to large gradients in the image while returning values of zero in uniform regions. There is an extremely large number of edge detection operators available, each designed to be sensitive to certain types of edges.

There are many ways to perform edge detection. However, the majority of different methods may be grouped into two categories: Gradient and Laplacian. The visual comparison of the most commonly used Gradient, Laplacian and Gaussian distribution based edge detection techniques was analyzed and performed.

In Section 2, the problem of formulation with the Gradient, Laplacian and Gaussian distribution working methods is presented. In Section 3, the various edge detection techniques have been studied and analyzed. Section 4 Studies the geometric properties of Surfaces. Section 5 discusses the advantages and disadvantages of various edge detection techniques by visual comparisons that have been done by developing software in MATLAB 7.11. Section 6 discusses the conclusions reached by analysis and visual comparison of various edge detection techniques developed using MATLAB 7.11.

PROBLEM OF FORMULATION

There are problems of false edge detection, missing true edges, producing thin or thick lines and problems due to noise, etc. Suppose we have 1D-image, with an edge shown by the jump in intensity (ramp edge) as shown in Figure 1(a).

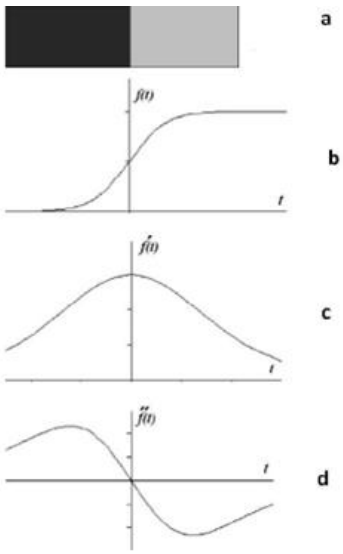

Figure 1: Image (a) with Function F(t) of Pixel Intensity Values

The signal (function) for image intensity is shown in Figure 1(b). The gradient (1st derivative) with respect to this signal is shown in Figure 1(c). Clearly, Figure 1(c) shows a maximum located at the center of the edge in the original signal. A pixel location is declared an edge location if the value of the gradient exceeds some threshold. As mentioned before, edges will have higher pixel intensity values than those surrounding them. So once a threshold is set, you can compare the gradient value to the threshold value and detect an edge whenever the threshold is exceeded [4, 10, 16]. Furthermore, when the first derivative is at a maximum, the second derivative is zero. As a result, another alternative to find the location of an edge is to locate the zeros in the second derivative. This method is known as the Laplacian (the second derivative of the signal) as shown in Figure 1(d) [4, 16].

Now let me present the definitions of Gradient, Laplacian and Gaussian distribution in 2D-image

GRADIENT [7, 16, 20]

DEFINITION 2.1 In 2-dimension the gradient of function f(x,y) is:

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix} \quad , \quad |\nabla f| = \sqrt{f_x^2 + f_y^2} \quad , \quad \theta = \tan^{-1}\left(\frac{f_y}{f_x}\right) \tag{1}$$

Where $|\nabla f|$ is called magnitude of gradient and θ its direction.

The gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image. Let $I(x, y)$ is the function define pixel intensity values of an image I (Note $I(x, y)$ is discrete function)

$$\begin{aligned} \frac{\partial I}{\partial x} &= \lim_{h_x \to 0} \frac{I(x+h_x, y) - I(x,y)}{h_x} = I(x+1, y) - I(x,y), \\ \frac{\partial I}{\partial y} &= \lim_{h_y \to 0} \frac{I(x, y+h_y) - I(x,y)}{h_y} = I(x, y+1) - I(x,y) \end{aligned} \tag{2}$$

Where $h_x = 1, h_y = 1$ because that I is discrete function.

Using pixel-coordinate notation (note: j corresponds to the x direction and i to the negative y direction, as shown in Figure 2) we obtain the following:

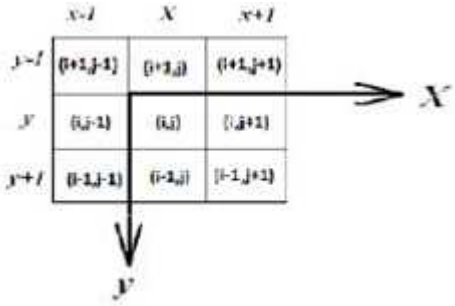

Figure 2: Pixel-Coordinate Notation

$$\frac{\partial I}{\partial x} = I(i, j+1) - I(i, j) \quad , \quad \frac{\partial I}{\partial y} = I(i-1, j) - I(i, j) \tag{3}$$

Definition 2.2 A kernel is a small matrix whose values are called weights. Each kernel has an origin, which is usually one of its positions.

The two kernels of Gradient in direction x and y are shown in Table 1

Table 1: The Two Kernels of Gradient

$$K_x = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & -1 & 1 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \quad , \quad K_y = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & -1 & 0 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

Now we can find $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ as follow

$$\frac{\partial I}{\partial x} = I_x = I * K_x \quad , \quad \frac{\partial I}{\partial y} = I_y = I * K_y \tag{4}$$

The gradient magnitudes are sometimes simplified by applying Manhattan distance measure as $|\nabla I| = |I_x| + |I_y|$ to reduce the computational complexity.

LAPLACIAN [7, 16, 20]

Definition 2.3 In 2-dimension the Laplacian of function f(x,y) is:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{5}$$

The Laplacian method searches for zero crossings in the second derivative of the image to find edges. To find the Laplacian of an image with pixel intensity values $I(x, y)$ we first find second derivative operators oriented in the x and y directions. From the equations (2) we get

$$\frac{\partial^2 I}{\partial x^2} = I(x+2,y) - 2I(x+1,y) + I(x,y),$$
$$\frac{\partial^2 I}{\partial y^2} = I(x,y+2) - 2I(x,y+1) + I(x,y) \quad (6)$$

we can be writing (6) as

$$\frac{\partial^2 I}{\partial x^2} = I(x+1,y) - 2I(x,y) + I(x-1,y),$$
$$\frac{\partial^2 I}{\partial y^2} = I(x,y+1) - 2I(x,y) + I(x,y-1) \quad (7)$$

Using pixel-coordinate notation in Figure 2 we obtained on

$$\frac{\partial^2 I}{\partial x^2} = I(i,j+1) - 2I(i,j) + I(i,j-1),$$
$$\frac{\partial^2 I}{\partial y^2} = I(i+1,j) - 2I(i,j) + I(i-1,j) \quad (8)$$

thus, the discrete Laplacian of an image with pixel intensity values $I(x,y)$ is:

$$\nabla^2 I = -4I(i,j) + I(i,j+1) + I(i,j-1) + I(i+1,j) + I(i-1,j) \quad (9)$$

A discrete Laplacian kernel is yielded by simply adding 1D second derivative operators oriented in the x and y directions as shown in Table 2

Table 2: The Kernel of Laplacian by Adding the X and Y Directions

0	0	0		0	1	0		0	1	0
1	-2	1	+	0	-2	0	=	1	-4	1
0	0	0		0	1	0		0	1	0

By also adding 1D second derivative operators oriented diagonally, the most common kernel of the discrete Laplacian is shown in. Table 3

Table 3: The Kernel of Laplacian by Adding the X, Y and Diagonals Directions

0	1	0		1	0	0		0	0	1		1	1	1
1	-4	1	+	0	-2	0	+	0	-2	0	=	1	-8	1
0	1	0		0	0	1		1	0	0		1	1	1

GAUSSIAN DISTRIBUTION [20]

Because kernels are used in gradient or laplacian are approximating a first or second derivative measurement on the image, they are very sensitive to noise. To counter this, the image is often Gaussian filtered before applying Gradient or Laplacian in edge detection.

Definition 2.4 In 2-dimension, the Gaussian distribution function is centered on zero with standard deviation σ has the form:

$$G(x,y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{10}$$

as shown in Figure 3.

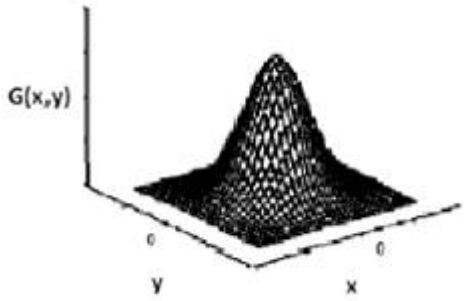

Figure 3: Gaussian Distribution Function

Gaussian filtering is used to remove noise from an image. The kernel of a Gaussian filter with a standard deviation of $\sigma = 1.4$ is shown in Table 4

Table 4: The Kernel of Gaussian Filtering with $\sigma = 1.4$

$$K_G = \frac{1}{159}$$

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	4
2	4	5	4	2

EDGE DETECTION METHODS

Classical Methods

These methods are based on gradient in edge detection. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point. A thresholding strategy [11] is used for determining the local maxima in the gradient image to determine edge points. Pixels at which the magnitude of the gradient is above a certain threshold T can be considered to be edge pixels. The edge map E is given by

$$E[m,n] = \begin{cases} 1, & |\nabla I(m,n)|) \geq T \\ 0, & otherwise. \end{cases} \tag{11}$$

General Algorithm [20] used in edge detection by classical operators is:

- Smooth the input image $I\hat{}(x,y) = I(x,y) * K_G(x,y)$
- $I_{\hat{x}} = I\hat{}(x,y) * K_x(x,y)$
- $I_{\hat{y}} = I\hat{}(x,y) * K_y(x,y)$

- $|\nabla I\ \hat{(}x,y)|=|I_{\hat{x}}|+|I_{\hat{y}}|$
- $\theta = tan^{-1}(I_{\hat{y}}/I_{\hat{x}})$
- If $|\nabla I\ \hat{(}x,y)|> T$, then possible edge point.

where : T is threshold , $K_G(x,y)$ is kernel of the Gaussian filtering , K_x and K_y are kernels of operator edge detection.

Robert's Cross Operator [12, 18]

The Roberts Cross operator performs a simple and quick computing, 2-D spatial gradient measurement on an image. The operator consists of a pair of 2×2 convolution kernels as shown in Table 5

Table 5: The Two Kernels of Robert's Operator

$$K_x = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array} \quad , \quad K_y = \begin{array}{|c|c|}\hline 0 & -1 \\ \hline 1 & 0 \\ \hline \end{array}$$

Remark: The kernel K_x is simply and the other is obtained by rotating K_x by angle 90°.

These kernels are designed to respond maximally to edges running at 45° to the pixel grid, one kernel for each of the two perpendicular orientations. The gradient magnitude is given by steps 1-4 in The previous algorithm and the angle of orientation of the edge (relative to the pixel grid orientation) is given by $\theta = tan^{-1}(\frac{I_{\hat{y}}}{I_{\hat{x}}}) - \frac{3\pi}{4}$.

Prewitt Operator [7, 14]

The operator consists of a pair of 3×3 convolution kernels, as shown in Table 6

Table 6: The Two Kernels of Prewitt Operator

$$K_x = \begin{array}{|c|c|c|}\hline -1 & 0 & +1 \\ \hline -1 & 0 & +1 \\ \hline -1 & 0 & +1 \\ \hline \end{array} \quad , \quad K_y = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 0 & 0 & 0 \\ \hline +1 & +1 & +1 \\ \hline \end{array}$$

Remark: The kernel K_x is simply and the other is obtained by rotating K_x by angle 90°.

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The gradient magnitude is given by steps 1-4 in The previous algorithm and the angle of orientation of the edge (relative to the pixel grid orientation) is given by $\theta = tan^{-1}(\frac{I_{\hat{y}}}{I_{\hat{x}}})$

Sobel Operator [12, 14, 15]

Sobel operator is similar to the Prewitt operator and is used for detecting vertical and horizontal edges in images.

This operator has a pair of 3×3 convolution kernels, as shown in Table 7

Table 7: The Two Kernels of Sobel Operator

$$K_x = \begin{array}{|c|c|c|}\hline -1 & 0 & +1 \\ \hline -2 & 0 & +2 \\ \hline -1 & 0 & +1 \\ \hline \end{array} \quad , \quad K_y = \begin{array}{|c|c|c|}\hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline +1 & +2 & +1 \\ \hline \end{array}$$

Canny Edge Detection Algorithm [5, 14]

This operator is also based on gradient in edge detection. The Canny edge detection algorithm is known to many as the optimal edge detector.

The General Algorithm [5, 20] used in the edge detection by Canny method is :

- Smooth the image with a Gaussian filter to reduce noise and unwanted details and textures $\hat{I}(i, j) = I(i, j) * G(i, j)$.
- Compute $\hat{I}_i(i, j)$, $\hat{I}_j(i, j)$ and then gradient of $\hat{I}(i, j)$ by using any of the gradient operators (roberts, Prewitt, Sobel) to get: $M(i, j) = \sqrt{\hat{I}_i^2(i, j) + \hat{I}_j^2(i, j)}$
- Threshold M: $M_T(i, j) = \begin{cases} M(i, j), & if\ M(i, j) > T; \\ 0, & otherwise\ 0. \end{cases}$
- Suppress non-maxima pixels in the edges in M_T obtained above to thin the edge ridges (as the edges might have been broadened in step 1). To do so, check to see whether each non-zero $M_T(i, j)$ is greater than its two neighbors along the gradient direction $\theta(i, j)$. If so, keep $M_T(i, j)$ unchanged, otherwise, set it to 0.
- Threshold the previous result by two different thresholds T_1 and T_2 (where $T_1 < T_2$) to obtain two binary images T_1 and T_2. Note that compared to T_1, T_2,they have less noise and fewer false edges but larger gaps between edge segments.
- Link edge segments in T_2 to form continuous edges. To do so, trace each segment in T_2 to its end and then search its neighbors in T_1 to find any edge segment in T_1 to bridge the gap until reaching another edge segment in T_2.

LAPLACIAN OPERATOR

Zero Crossing [7, 20]

A thresholding strategy [11] is used for determining the zero crossings in the Laplacian image to determine edge points. At a zero crossing, a pixel on one side of the zero-crossing is positive and negative on the other side. The greater the difference between these two pixels is at a point, the more likely the zero crossing corresponds to an edge. Therefore, in a digital image, pixels at which this difference is above a certain threshold T can be considered to be edge pixels.

The edge map E is given by

$$E[m,n] = \begin{cases} 1, & L(x, y) \geq T \\ 0, & otherwise. \end{cases} \quad (12)$$

Laplacian of Gaussian [14, 17, 20]

We can convolve the Gaussian smoothing filter with the Laplacian filter first of all, and then convolve this hybrid filter with the image to achieve the required result. Doing things this way has two advantages: Since both the Gaussian and the Laplacian kernels are usually much smaller than the image, this method usually requires far fewer arithmetic operations. The Laplacian of Gaussian (LoG) kernel can be pre-calculated in advance so only one convolution needs to be performed at run-time on the image. The 2-D LoG function centered on zero and with Gaussian standard deviation ó has the form:

$$LoG(x, y) = -1/\pi\sigma^4[1-(\frac{x^2+y^2}{2\sigma^2})]e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (13)$$

as shown in Figure 4. Where The x and y axes are marked in standard deviations σ.

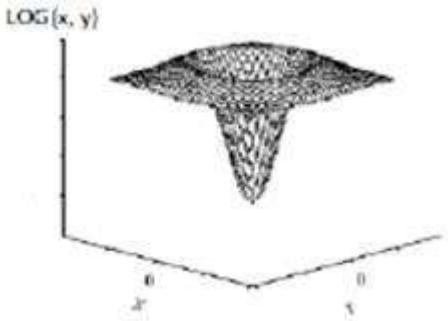

Figure 4: 2-D Log Function

The kernel of The LoG (for standard deviation $\sigma < 0.5$) is shown in Table 8

Table 8: The Kernel of LOG Operator with $\sigma < 0.5$

$$K_{LOG} = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

Note that kernel size increases with σ as in Gaussian filtering [8, 9, 14].

STUDY THE GEOMETRIC PROPERTIES OF SURFACES

As an application, we study the geometric properties of the following surfaces

Revolution Surface: consider a Revolution surface represented by the parametric equation

$$R(u^1, u^2) = (u^1 \cos u^2, u^1 \sin u^2, e^{u^1}), u^1 \in (-3,3) \quad (14)$$

as shown in Figure 5(a).

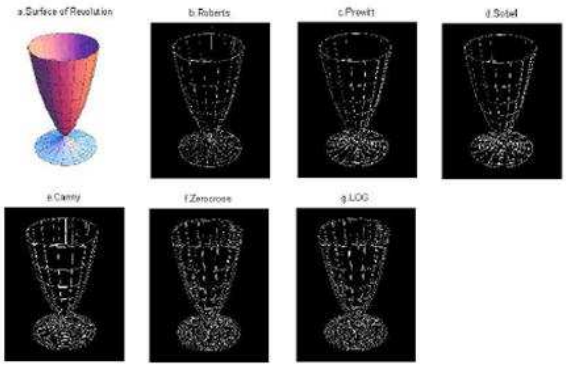

Figure 5: Revolution Surface and its Edge by Various Edge Detection Methods

To describe the revolution surface geometric [6, 13] we define parametric lines as follows: $u^2 = const.$ is called meridians of Revolution surface. While $u^1 = const.$ is called parallels of Revolution surface. And by calculating the tangents R_1, R_2 of meridians and parallels on revolution surface we obtain

$$R_1 = (\cos u^2, \sin u^2, e^{u^1}), R_2 = (-u^1 \sin u^2, u^1 \cos u^2, 0) \tag{15}$$

The first fundamental covariant quantities $g_{\alpha\beta}$ and contravariant quantities $g^{\alpha\beta}$ are given form $g_{\alpha\beta} = < R_\alpha, R_\beta >$, $g_{\alpha\beta} \cdot g^{\beta\gamma} = \delta_\alpha^\gamma$ Explicitly

$$g_{11} = 1 + e^{2u^1}, g_{12} = g_{21} = 0, g_{22} = (u^1)^2, g = (u^1)^2(1 + e^{2u^1}) \tag{16}$$

$$g^{11} = \frac{g_{22}}{g} = \frac{1}{\zeta_1}, g^{12} = \frac{-g_{21}}{g} = 0 = g^{21}, g^{22} = \frac{g_{11}}{g} = \frac{1}{(u^1)^2} \tag{17}$$

Where $\zeta_1 = 1 + e^{2u^1}$

The unit normal vector field N on (14) is given from

$$N(u^1, u^2) = \frac{< R_1, R_2 >}{\sqrt{g}} = \frac{1}{\sqrt{\zeta_1}}(-e^{u^1} \cos u^2, -e^{u^1} \sin u^2, 1) \tag{18}$$

The second partial derivatives for (14) are

$$R_{11} = (0, 0, e^{u^1}), R_{12} = (-\sin u^2, \cos u^2, 0), R_{22} = (-u^1 \cos u^2, -u^1 \sin u^2, 0) \tag{19}$$

The second fundamental quantities $L_{\alpha\beta} =< R_{\alpha\beta}, N >$ take the form

$$L_{11} = \frac{e^{u^1}}{\sqrt{\zeta_1}}, L_{12} = L_{21} = 0, L_{22} = \frac{u^1 e^{u^1}}{\sqrt{\zeta_1}} \tag{20}$$

Gaussian curvature K and mean curvature H and principal curvatures k_1, k_2 are given from

$$K = k_1 k_2 = \frac{L}{g} = \frac{L_{11}L_{22} - L_{12}^2}{g_{11}g_{22} - g_{12}^2} = \frac{e^{2u^1}}{u^1 \zeta_1^2} \tag{21}$$

$$2H = k_1 + k_2 = g^{\alpha\beta} L_{\alpha\beta} = g^{11}L_{11} + 2g^{12}L_{12} + g^{22}L_{22} = \frac{e^{u^1}\eta_1}{u^1 \zeta_1^{3/2}} \tag{22}$$

Where $\eta_1 = 1 + u^1 + e^{2u^1}$. From the equations(21), (22) we get

$$k_1 = \frac{e^{u^1}}{2}\sqrt{\frac{\xi_1^2}{(u^1)^2\zeta_1^3}} + \frac{e^{u^1}\eta_1}{2u^1\zeta_1^{3/2}} \quad , \quad k_2 = -\frac{e^{u^1}}{2}\sqrt{\frac{\xi_1^2}{(u^1)^2\zeta_1^3}} + \frac{e^{u^1}\eta_1}{2u^1\zeta_1^{3/2}} \tag{23}$$

Where $\xi_1 = 1 - u^1 + e^{2u^1}$

Saddle Surface (Ruled Surface): consider a saddle surface represented by the parametric equation (as shown in Figure 6(a))

$$R(u^1, u^2) = (\frac{1}{2}(u^1 + u^2), \frac{1}{2}(u^2 - u^1), u^1 u^2) \tag{24}$$

To describe the Saddle surface geometric [6, 13] we define parametric lines as follows: $u^2 = const.$ is called meridians of Saddle surface. While $u^1 = const.$ is called parallels of Saddle surface. And by calculating the tangents R_1, R_2 of meridians and parallels on Saddle surface we obtain

$$R_1 = (\frac{1}{2}, -\frac{1}{2}, u^2),\ R_2 = (\frac{1}{2}, \frac{1}{2}, u^1) \tag{25}$$

The first fundamental covariant quantities $g_{\alpha\beta}$ and contravariant quantities $g^{\alpha\beta}$ are given form $g_{\alpha\beta} = < R_\alpha, R_\beta >$, $g_{\alpha\beta} \cdot g^{\beta\gamma} = \delta_\alpha^\gamma$ Explicitly

$$g_{11} = \frac{1}{2} + (u^2)^2, g_{12} = g_{21} = u^1 u^2, g_{22} = \frac{1}{2} + (u^1)^2, g = \frac{1}{4}\zeta_2 \tag{26}$$

where $\zeta_2 = 2(u^1)^2 + 2(u^2)^2 + 1$

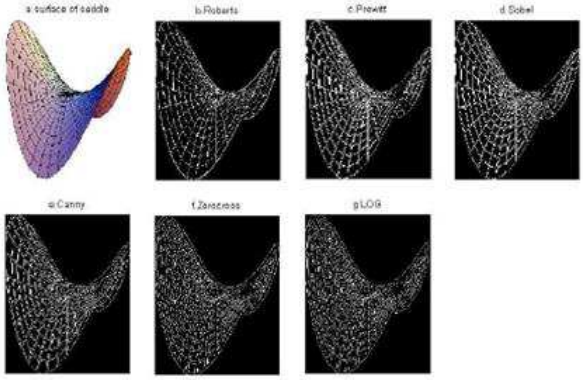

Figure 6: Saddle Surface and its Edge by Various Edge Detection Methods

$$g^{11} = \frac{2+4(u^1)^2}{\zeta_2}, g^{12} = \frac{-4u^1u^2}{\zeta_2} = g^{21}, g^{22} = \frac{2+4(u^2)^2}{\zeta_2} \tag{27}$$

The unit normal vector field N on the saddle surface (24) is given from

$$N(u^1,u^2) = \frac{< R_1, R_2 >}{\sqrt{g}} = \frac{1}{\sqrt{\zeta_2}}(-u^1 - u^2, u^2 - u^1, 1) \tag{28}$$

The second partial derivatives for (24) are

$$R_{11} = (0,0,0), R_{12} = (0,0,1) = R_{21}, R_{22} = (0,0,0) \tag{29}$$

The second fundamental quantities $L_{\alpha\beta} =< R_{\alpha\beta}, N >$ take the form

$$L_{11} = 0, L_{12} = L_{21} = \frac{1}{\sqrt{\zeta_2}}, L_{22} = 0, L = \frac{-1}{\zeta_2} \tag{30}$$

Gaussian curvature K and mean curvature H and principal curvatures k_1, k_2 are given from

$$K = k_1k_2 = \frac{L}{g} = \frac{-4}{\zeta_2^2} \neq 0 \tag{31}$$

$$2H = k_1 + k_2 = g^{\alpha\beta}L_{\alpha\beta} = -\frac{8u^1u^2}{\zeta_2^{3/2}} \tag{32}$$

by solving the two equations(31), (32) we get

$$k_1 = 2\sqrt{\xi_2} - \frac{4u^1u^2}{\zeta_2^{3/2}}, \quad k_2 = -2\sqrt{\xi_2} - \frac{4u^1u^2}{\zeta_2^{3/2}} \tag{33}$$

where $\xi_2 = \frac{\eta_2}{\zeta_2^3}$, $\eta_2 = (1+2(u^1)^2)(1+2(u^2)^2)$

ADVANTAGES AND DISADVANTAGES OF VARIOUS EDGE DETECTION TECHNIQUES

Edge detection of all two types (Revolution and Saddle surfaces) was performed on the above Figures 5(a)-5(g) and 6(a)-6(g). Canny yielded the best results. This was expected as Canny edge detection accounts for regions in an image. Canny yields thin lines for its edges by using non-maximal suppression. Canny also utilizes hysteresis with thresholding.

As edge detection is a fundamental step in computer vision, it is necessary to point out the true edges to get the best results from the matching process. That is why it is important to choose edge detectors that fit best to the application. In this respect, we present some advantages and disadvantages of edge detection techniques [14, 16, 19] as follow

- **Classical Methods (Roberts, Prewitt, Sobel)**
  - **Advantages:** Simplicity, Detection of edges and their orientations.
  - **Disadvantages:** Sensitivity to noise, Inaccurate.
- **Zero-Crossing (Laplacian)**
  - **Advantages:** Detection of edges and their orientations. Having fixed characteristics in all directions.
  - **Disadvantages:** Responding to some of the existing edges, Sensitivity to noise.
- **Laplacian of Gaussian(LoG)**
  - **Advantages**: Finding the correct places of edges, Testing wider area around the pixel.
  - **Disadvantages:** Malfunctioning at the corners, curves and where the gray level intensity function varies. Not finding the orientation of edge because of using the Laplacian filter.
- **Gaussian (Canny)**
  - **Advantages:** Using probability for finding error rate, Localization and response. Improving signal to noise ratio, Better detection specially in noise conditions.
  - **Disadvantages:** Complex Computations, False zero crossing, Time consuming.

CONCLUSIONS

Since edge detection is the initial step in object recognition (as surfaces geometric), it is important to know the differences between edge detection techniques. In this paper we studied the most commonly used edge detection techniques of Gradient-based and Laplacian-based edge detection for geometric surfaces (Revolution surface and Ruled surface) as shown in Figures 5(a)-5(g) and 6(a)-6(g). The software is developed using MATLAB 7.11.

Gradient-based algorithms such as the Prewitt filter have a major drawback of being very sensitive to noise. The size of the kernel filter and coefficients are fixed and cannot be adapted to a given image. An adaptive edge-detection algorithm is necessary to provide a robust solution that is adaptable to the varying noise levels of these images to help distinguish valid image contents from visual artifacts introduced by noise. The performance of the Canny algorithm depends heavily on the adjustable parameters, which are the standard deviation for the Gaussian filter, and the threshold values, T_1 and T_2. also controls the size of the Gaussian filter. The bigger the value, the larger the size of the Gaussian filter becomes. This implies more blurring, necessary for noisy images, as well as detecting larger edges. As expected,

however, the larger the scale of the Gaussian, the less accurate is the localization of the edge. Smaller values of ó imply a smaller Gaussian filter which limits the amount of blurring, maintaining finer edges in the image. The user can tailor the algorithm by adjusting these parameters to adapt to different environments. Canny's edge detection algorithm is computationally more expensive compared to Sobel, Prewitt and Robert's operator. However, the Canny's edge detection algorithm performs better than all these operators under almost all scenarios. Evaluation of the surface images showed that under noisy conditions, Canny, LoG, Laplacian, Sobel, Prewitt, Roberts's exhibit better performance, respectively.

Nassar et.al. studied the image processing problem using the ridge and Ravens on the surface through the maximum and minimum of the principal curvature [1, 2, 3]. In a forthcoming paper we will find method to edge detection of surface-image based on geometric properties of surfaces, where we will try to overcome flaws in previous methods.

REFERENCES

1. N. H. Abdel-All and A. A. Al-moneef, Ridges and Singularities on Hypersurfaces. *Jour. of Inst. of maths.* & *Comp. Sciences* 21 (2008) 109-116.
2. N. H. Abdel-All and A. A. Al-moneef, Generalized Ridgea and Raines on an Equiform Motion. *Studies in Mathmatical sciences,* 4 (2012) 76-85.
3. N. H. Abdel-All and A. A. Al-moneef, Local Study of Sigularities on an Equiform Motion. *Studies in Mathmatical sciences,* 5 (2012) 26-36.
4. F. Bergholm, Edge focusing. In: Proceedings of 8th International Conference on Pattern Recognition. *Paris, France,* (1986) 597-600.
5. J. Canny, A Computational Approach to Edge Detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* 8(1986) 679-698.
6. M. d. Carmo, Differential Geometry of Curves and Surfaces, *Prentice-Hall, Englewood Cliffs,* 1976.
7. R.C. Gonzalez and R. E. Woods, Digital Image Processing, *Addison - Wesley Publishing Company,* 1992.
8. W. E. Grimson, E. C. Hildreth, Comments on digital step edges from zero crossings of second directional derivatives. *IEEE Trans. Pattern Anal. Machine Intell,* 7(1985) 121-129.
9. R. M. Haralick, Digital step edges from zero crossing of the second directional derivatives. *IEEE Trans. Pattern Anal. Machine Intell,* 6(1984) 58-68.
10. M. Kaushal, A. Singh, B. Singh, EAdaptive Thresholding for Edge Detection in Gray Scale Images. *IET Bhaddal. Punjab,* 2(2010) 2077-2082.
11. Khalil, A. Aggoun, A. ELmabrouk, On Edge Detector Using Local Histogram Analysis, *SPIE Visual Communications and Image Processing,* 5150(2003) 2141-2151.
12. S. Lakshmi and V. Sankaranarayanan, A study of Edge Detection Techniques for Segmentation Computing Approaches. *IJCA, CASCT,* (2010) 35-41.
13. M. M. Lipschutz, Theory and problems of differential geometry, *McGraw-Hill, New York,* 1969.

14. R. Maini and H. Aggarwal, Study and Comparison of Various Image Edge Detection Techniques. *International Journal of Image Processing (IJIP),* 3(2009) 1-12.

15. J. Matthews, An introduction to edge detection: The sobel edge detector. *Available at http://www.generation5.org/ content/2002/im01.asp,* 01(2002).

16. V. G. Narendra and K. S. Hareesh, Study and Comparison of Various Image Edge Detection Techniques Used in Quality Inspection and Evaluation of Agricultural and Food Products by Computer Vision. *Int. J. Agric. Biol. Eng.* 4(2011) 83-90.

17. T. A. Poggio, On edge detection. *IEEE Trans. Pattern Anal. Machine Intell,* 8(1986) 163-187.

18. L. G. Roberts, Machine perception of 3-D solids-series. Optical and Electro-Optical Information Processing. *MIT Press, Cambridge, Mass.,* (1965) 159-197.

19. M. Sharifi, M. Fathy, M. T. Mahmoudi, A Classified and Comparative Study of Edge Detection Algorithms. *Coding and Computing IEEE,* 2(2000).

20. E. Trucco and A. Verri, Introductory Techniques for 3-D Computer Vision. *Prentice Hall,* 1998.